Remarks

Initially, Applicant would like to thank the Examiner for the teleconference of February 13, 2007. Applicant has amended claims 2, 4-9, 11-15, 25, 43-44 and 47-48; and has cancelled claims 1, 3, 16-24, 34-42 and 46. Applicant respectfully submits that no new matter was added by the amendment, as all of the amended matter was either previously illustrated or described in the drawings, written specification and/or claims of the present application. Entry of the amendment and favorable consideration thereof is earnestly requested.

Also, as amended, all claims require translating source code from a first computer language to source code in a second computer language. Applicant respectfully submits that U.S. Patent 6,314,429 ("Simser") fails to teach, suggest or disclose this. Rather, Simser is directed toward a system for providing a "bi-directional conversion library" and specifically "a method for translating data structures used by a computer program written in a first computer language to equivalent data structures in a second computer language." (Col. 1, lns. 7-10; Col. 6, ln. 21.) This is very different from the presently claimed invention, which is directed toward translating source code and the functionality therewith, not simply to converting data used by a first program in a first language to data that may be used by a second program in a second language. Accordingly, Simser does not and can not perform translation of first computer language source code to second computer language source code as required by the pending claims. This being the case, Applicant notes that Simser fails to teach, or suggest an emulation Application Programming Interface library having a table including equivalent functions in a second computer language as required by all the claims. Rather, as stated above, Simser is directed to transforming data structures used by disparate programs, not to translating the programs themselves. Accordingly, Simser fails to teach or suggest mapping first source code to a table having "equivalent functions" in a second source code.

All claims require an emulation Application Programming Interface library. Andrews et al. fails to teach or disclose this limitation. For example, while Andrews et al. uses the term "emulation", it never uses this term in connection with an Application Programming Interface library. Rather, Andrews et al. teaches use of emulation macros for emulating language features in the source language. (Col. 5, ln. 8; Col. 13, lns. 49-60.) This does not allow for context-insensitive translation, which is a requirement for using an API emulation library and for supporting bi-directional translation. (*See e.g.*, Col. 6, lns 17-20; Col. 7, lns. 4-23 & 28-41; Col. 14, ln. 66 – Col. 15, ln. 7; Col. 15, lns. 14-21.) Accordingly, Andrews et al. actually teaches away from the presently pending claims.

In addition, as amended, all claims require translating a first object-oriented programming ("OOP") computer language source code to a second OOP computer language source code. (*See*, Par. 4.) While U.S. Patent No. 5,768,564 (Andrews et al.) is directed to a computer language translation system, Andrews et al. fails to teach translation of OOP computer language source code. For example, Andrews et al. teaches that the system disclosed therein "does not generate object-oriented code." (Col. 5, lns. 5-6.) Accordingly Andrew et al. can not anticipate any of the pending claims. *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). In fact, Andrews actually teaches directly away from this limitation. (*See e.g.*, Col. 5, lns. 5-12.)

Applicant further notes that claims 14, 25 and 43 still further include the limitation that the second computer language source code can independently provide the equivalent type of data manipulations provided in said first computer language source code without reference to the first computer language source code. This is clearly not taught or suggested in Simser, which teaches that it is necessary to use both the first and the second computer languages when using the translated program. (*See*, FIG. 1 including use of both Java and Legacy and associated description.) Andrews et al. also fails to teach or suggest this limitation. For example, Andrews et al. teaches that "[u]nfortunately, converting a product to C++ does not immediately obviate the need to

use pTAL language." (Col. 11, lns 39-40.) In addition, Andrews et al. requires that memory layout of data in the second language be similar to that of the first language, which causes the second language source code to be dependent on the first language source code as well as the runtime environment of the first language. (Col. 13, ln. 66 – Col. 14, ln. 8.)

Accordingly, neither Simser nor Andrews et al. teach, disclose or suggest that the second computer language source code operate independently or without reference to the first computer language source code, but in fact, both references teach a co-dependency on programs in both languages.

Applicant still further submits that claim 47 requires that the second OOP computer language source code performing the data manipulation without use of a virtual machine, while claims 48 requires that the second OOP computer language source code performing the data manipulation without use of a garbage collector. Again, both Simser and Andrews et al. require reference to both languages for the systems to work. However, as claimed in the present invention, the generated second computer language source code is completely independent from the first computer language and does not need to reference it or utilize it once the second computer language is generated. This includes, for example, not needing the use of a virtual machine or a garbage collector used by the first computer language.

Applicants further note with regard to the cited references that, all claims re directed toward systems that translate a first OOP computer language source code to a second OOP computer language source code. Andrews et al. does not actually translate to industry-standard computer languages, but instead requires proprietary extensions to the target computer language. For example, "[t]he target language was extended to support some special functional requirements of Tandem systems software . . . For example, pragmas were added to produce similar data layout between C++ and pTAL." (Col. 13, ln. 66 – Col. 14, ln. 8.) This prevents the Andrews approach from being

used with preexisting programming tools, such as ANSI/ISO C++ compilers. In contrast, the present system produces translated programs that do not require proprietary computer language extensions or proprietary compilers as per claims 47 and 48.

Still further, claims 2 and 15 require a bi-directional translator. Andrews et al. does not translate unused code. (Col. 7, lns. 22-23.) This makes Andrews et al. unsuitable for both lossless unidirectional translation and bi-directional translation. In addition, Andrews replicates and reorders source code. (Col. 9, lns. 29-35.) This approach does not preserve the form of expression in the original source code making both lossless unidirectional translation and bi-directional translation impossible. Finally, Andrews et al. "generates idioms in the target language" instead of literal translation of source code constructs. (Col. 8, lns. 1-3). Again, this approach does not preserve the form of expression in the original source code and thwarts bi-directional translation.

It is respectfully submitted that claims2, 4-15, 25-33, 43-45 and 47-48, all of the claims remaining in the application, are in order for allowance and early notice to that effect is respectfully requested.

Respectfully submitted,

March 1, 2007

/Wesley W. Whitmyer, Jr./

Wesley W. Whitmyer, Jr., Registration No. 33,558
Steven B. Simonis, Registration No. 54,449
Attorneys for Applicant
ST. ONGE STEWARD JOHNSTON & REENS LLC
986 Bedford Street
Stamford, CT 06905-5619
203 324-6155